Closure properties of RE, Rec

Recall that a language L is
- **recursively enumerable** (RE) if there exists a TM for L,
- **recursive** (Rec) if there exists a TM for L which halts on every input (i.e. also on strings not from L).

Is the class RE closed under union? And intersection? **YES**

Is the class Rec closed under union? And intersection? **YES**

What can we say about complement? **we'll see**
**Closure properties of RE, Rec**

**Lemma**: RE languages are closed under union.

I.e. given two TMs $M_1, M_2$, describe a TM $M_3$ s.t. $L(M_3) = L(M_1) \cup L(M_2)$

**Idea**: $M_3$ runs both $M_1, M_2$ simultaneously.

- we can assume that $M_3$ has 2 tapes
- on the 1st tape it simulates $M_2$
- on the 2nd tape it simulates $M_2$
- the first step is $M_1$, the second $M_2$, the third $M_1$, the forth $M_2$, etc. ← alternates $M_1, M_2$
- if either $M_1$ or $M_2$ accepts, $M_3$ accepts

**Lemma**: RE languages are closed under intersection.

**Idea**: same as above, except that $M_3$ must wait until both $M_1, M_2$ accept

**Note**: for intersection we can also do this:

1) copy input to store it
2) run $M_1$
3) if accepts, copy input back and clear the rest of the tape
4) run $M_2$
5) if accepts, accept
Lemma: Recursive languages are closed under union.

Same idea as for RE works here as well.

Now we have to say that $M_3$ never gets to an infinite loop.

$\Rightarrow$ This follows from $M_1, M_2$ never getting to an infinite loop.

Lemma: Recursive languages are closed under intersection.

Similar as above and on the previous page.
Closure properties of RE, Rec

Lemma: Recursive languages are closed under complement.

Lemma: RE languages are not closed under complement.
Closure properties of RE, Rec

Thm : L and L' are RE iff L is recursive.
Noam Chomsky studied grammars as potential models for natural languages. He classified grammars according to these four types:

- **Type 0 Grammars**: Unrestricted Grammars (generate RE languages)
- **Type 1 Grammars**: Context-sensitive (monotone) Grammars (generate context-sensitive languages)
- **Type 2 Grammars**: Context-free Grammars (generate context-free languages)
- **Type 3 Grammars**: Regular Grammars (generate regular languages)
Def: An *unrestricted grammar* is a 4-tuple \( G=(V,\Sigma,S,P) \) where

- \( V \) is a finite set of variables
- \( \Sigma \) is a finite set of terminal symbols
- \( S \in \Sigma \) is the start symbol
- \( P \) is a finite set of productions of the form \( \alpha \rightarrow \beta \) where
  \( \alpha \in (V \cup \Sigma)^+ \) and \( \beta \in (V \cup \Sigma)^* \)

\( (V \text{ and } \Sigma \text{ are assumed to be disjoint}) \)
Unrestricted Grammars (Type 0)

Example: Give an unrestricted grammar for \( \{ a^k b^k c^k \mid k \geq 0 \} \)
Example: Give an unrestricted grammar for \( \{ a^j \mid j = 2^k, k \geq 0 \} \)
Context-sensitive Gram. (Type 1)

Def: A type 0 grammar $G=(V, \Sigma, S, P)$ is context-sensitive if for every production rule $\alpha \rightarrow \beta$ in $P$, $|\alpha| \leq |\beta|$.

Which of our examples of type 0 grammars are context-sensitive?
Context-sensitive Gram. (Type 1)

Def: A type 0 grammar $G=(V, \Sigma, S, P)$ is **context-sensitive** if for every production rule $\alpha \rightarrow \beta$ in $P$, $|\alpha| \leq |\beta|$.

Lemma: Every context-free language which does not contain $\Lambda$ is context-sensitive.
**Context-sensitive Gram. (Type 1)**

**Def:** A type 0 grammar $G=(V,\Sigma,S,P)$ is **context-sensitive** if for every production rule $\alpha \rightarrow \beta$ in $P$, $|\alpha| \leq |\beta|$.

**Lemma:** Every context-free language which does not contain $\Lambda$ is context-sensitive.

**Def:** A linear-bounded automaton $A$ is a TM which never rewrites a blank to a non-blank symbol.

**Lemma:** A language $L$ is context-sensitive iff there exists a linear-bounded automaton accepting $L$. 

[Section 10.3]
Regular Grammars (Type 3)

Def: A type 0 grammar $G=(V,\Sigma,S,P)$ is **regular** if every production rule in $P$ is of the form $A \rightarrow \sigma B$ or $A \rightarrow \sigma$, where $A,B \in V$ and $\sigma \in \Sigma$.

Lemma: A language $L$ is regular iff there exists a regular grammar for $L-\{\Lambda\}$. 